Our Ref. No.: 82225P0189R

#17
LDS
3-7-01

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:) Examiner: David Y. Eng

Michael Powell, et al.)

Serial No.: 08/887,680)

Filed: July 3, 1997)

For: METHOD AND APPARATUS FOR EXTENDING COMPUTER ARCHITECTURE FROM 32 TO 64 BITS) Art Unit: 2783

Assistant Commissioner for Patents
Washington, D.C. 20231

APPEAL BRIEF

Dear Sir:

Applicants submit, in triplicate, the following Appeal Brief pursuant to 37 C.F.R. § 1.192 for consideration by the Board of Patent Appeals and Interferences. Applicants also submit herewith a check in the amount of $310.00 to cover the cost of filing the opening brief as required by 37 C.F.R. § 1.17(c). Please charge any additional amount due or credit any overpayment to deposit Account No. 02-2666.

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST

Michael Powell, the party named in the caption, Robert Cmelik, Shing Kong, David Ditzel, and Edmund Kelly transferred their rights to that which is disclosed in the subject application through an assignment recorded on December 21, 1990 (5555/0186) to Sun Microsystems, Inc. of Mountain View, California. Thus, as the owner at the time the brief is being filed, Sun Microsystems, Inc. of Mountain View, California is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences which will affect or be affected by the outcome of this appeal.

III. STATUS OF CLAIMS

Claims 1-38 are pending in this application. Claims 1-16 are allowed, but Claims 17-38 are rejected.

IV. STATUS OF AMENDMENTS

All amendments have been entered.

V. SUMMARY OF THE INVENTION

The invention relates to a method and apparatus for enabling a computer system to store from register files to memory, and restore from memory back to the register files, data from programs designed to operate in accordance with a first word size, as well as programs designed to operate in accordance with a second word size. This is accomplished by utilizing an indication to designate whether a particular procedure is using words of a first or second word size (Abstract). The indication is placed in at least one of a number of the registers of the processor designating the word size of the procedure (Col. 3, lines 10-12). In an embodiment, the indication is placed in the stack pointer because the system looks to the stack pointer to find the address of the save space. Thus, the save space for old procedures (i.e. thirty-two bit procedures) may utilize the same area of memory as would have been used under the old system software. Additional space may then be provided to store the registers of a sixty-four bit procedure (Col. 7, lines 2-13).

In an embodiment, the indication is placed in the most significant bit of the stack pointer (Col. 7, lines 27 and 28). Placing the indication in the most significant bit is advantageous in that the address in the stack pointer may be manipulated mathematically in addressing, and using the highest order bit allows the stack pointer address to be added to or subtracted from without affecting the indication (Col. 7, lines 39-44).

In an embodiment, the indication is also placed in the lowest order bit with a zero representing a thirty-two bit procedure and a one representing a sixty-four bit procedure (Col. 8, lines 43-49; 54-65; Col. 9, lines 6-16). Thus, the system can save and restore the register files in accordance with the procedure designated by the indication. Figures 4a and 4b further illustrate saving and restoring register files, including the use of the least significant bit (LS Bit) as an indication.

VI. ISSUES

The issues involved in this appeal are as follows:

Are Claims 17-38 unpatentable under 35 U.S.C. §251 as being an improper recapture of subject matter surrendered in the application for the parent upon which the present reissue is based?

Are Claims 17-38 unpatentable under 35 U.S.C. §112, first paragraph, as containing subject matter which was not described in the specification in such a way as to enable one skilled in the art to which it pertains, or with which it is most nearly connected, to make and/or use the invention?

Are Claims 33-38 unpatentable under 35 U.S.C. §112, second paragraph, as being incomplete for omitting essential steps, such omission amounting to a gap between the steps?

VII. GROUPING OF CLAIMS

Applicants submit that the claims do not stand or fall together. Accordingly, Applicants group the claims as follows:

Group I	Claim 17-32
Group II	Claim 33-38

The reason for the independent patentability of the separate groups is discussed in detail below.

VIII. ARGUMENT

The Examiner has rejected Claims 17-38 under 35 U.S.C. §251 as being an improper recapture. The Examiner has further rejected Claims 17-38 under 35 U.S.C. §112, first paragraph, as being non-enabling. The Examiner has rejected Claims 33-38 under 35 U.S.C. §112, second paragraph, as being incomplete for omitting essential steps.

A. Groups I and II: Rejection of Claims 17-38 Based on Improper Recapture

The Examiner argues that during prosecution of the parent application, Applicants relied upon two specific limitations to show patentability over the cited references (Preliminary Amendment, filed October 11, 1994, Application No. 08/321,459, Patent No. 5,430,864). Specifically, the Examiner points to "testing a most significant bit of the stack pointer register" and "setting a width indication bit". Both of these limitations will be addressed in turn.

1. "Testing a Most Significant Bit" v. "Testing a Least Significant Bit"

The Examiner suggests that recitation of the "least significant bit" (Claims 17-38) instead of the "most significant bit" (as recited in Claims 73-88 of the parent application) amounts to recapture of cancelled Claims 55-72 of the parent application because the combined scope of Claims 1-38 can be interpreted such that any bit can be tested within the stack point register. The

Examiner also notes that the finally rejected and cancelled Claims 55-72 of the parent application fail to specify which bit of the stack pointer register is tested. Applicants disagree with the Examiner's conclusion.

Applicants submit that in order for Claims 17-38 to run afoul of the recapture rule, they would necessarily recite "testing an indication bit" as recited in Claims 55-72 of the parent application. However, Claims 17-38 recite, among other limitations, only testing the least significant bit rather than "an indication bit" as set forth in Claims 55-72 of the parent application, and therefore, no impermissible recapture has occurred.

As the Examiner points out, cancelled Claims 55-72 of the parent application did not require any specific bit to be the indication bit, and thus, one could reasonably interpret Claims 55-72 of the parent application to read on a method which allowed any bit to be the indication bit. However, Applicants submit that the Examiner's interpretation of combined Claims 1-38 such that any bit can be tested is unreasonable. Only a cursory read is necessary to see that Claims 1-16 are directed towards embodiments in which only the most significant bit is tested while Claims 17-38 are directed towards embodiments in which only the least significant bit is tested. Each of Claims 1-38 recite one specific bit which can be tested as opposed to Claims 55-72 of the parent application which made no reference to any specific bit and could be construed to encompass any bit as the bit to be tested. Moreover, Applicants do not recite testing any bit other than either the most significant bit or the least significant bit in Claims 1-38. Thus, the Examiner's assertion that any bit can be tested, as recited by any individual claim or the entire claim set, is misplaced.

Accordingly, Applicants do not impermissibly recapture that which was given up by canceling Claims 55-72 of the parent application (i.e. ability to test any bit). Rather, Applicants have submitted claims which (1) are narrower than those that were given up, and (2) are directed to alternative, patentable embodiments in which a single, specific bit is tested. As such, "testing a least significant bit" should be patentable just as "testing a most significant bit" is patented.

2. Omission of "Setting a Width Indication Bit"

The Examiner contends that omitting "setting a width indication bit" from Claims 33-38 constitutes impermissible recapture of cancelled Claims 55-72 of the parent application because Claims 55-72 of the parent application did not include setting a width indication bit while Claims 73-88 of the parent application, which eventually issued, did include such a step. Applicants disagree with the Examiner's position.

A closer review of Claims 55-72 of the parent application shows that independent Claims 55 and 64 of the parent application both include a limitation analogous to setting the width indication bit. Specifically, Claims 55 and 64 of the parent application included limitations directed to "setting an indication bit corresponding to said procedure in said stack save area" (Claim 55, line 12; Claim 64, lines 12 and 13). This is analogous to "setting a width indication bit in the first stack save area" since both the "indication bit" of Claims 55 and 64 of the parent application and the "width indication bit" of Claims 73-88 of the parent application indicate whether the data being saved is of a first word size or a second word size. Thus, "setting a width indication bit" was not added in order to obtain the patent upon which the present reissue is based. Rather, the limitation was already present in an analogous form and was merely reworded in Claims 73-88.

Moreover, one need only look to the Advisory Action mailed on September 24, 1994, in connection with the parent application, to see that Examiner Phillip refused to enter the amendment (adding new Claims 73-88 of the parent application) because "testing a most significant bit of a stack pointer is a new limitation that would require further consideration and/or search". Absent from the Advisory Action is any characterization of the step of "setting the width indication bit" as a new limitation which would require further consideration and/or search. Obviously, Examiner Phillip failed to view "setting a width indication bit" as a new limitation, and thus, such a conclusion should not be reached in the present case.

In light of the foregoing, the absence of the step of "setting a width indication bit" is not an impermissible recapture of subject matter.

B. Groups I and II: Rejection of Claims 17-38 Based on a Disclosure That is Not Enabling

The Examiner contends that the specification, while being enabling for context switching by setting an indication at the most significant bit of a stack pointer register, does not reasonably provide for context switching by setting an indication at the least significant bit of a stack pointer register. Applicants disagree with the Examiner's conclusion.

In order to be enabling, the specification must teach one skilled in the art how to make and use the claimed invention. Applicants cancelled Claims 55-72 of the parent application and submitted Claims 73-88 (Preliminary Amendment, filed October 11, 1994), which eventually issued as Claims 1-16 of the parent application. Claims 73-88 of the parent application included a limitation of "testing a most significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure . . . each have a first word size or a second word size . . . ". This amendment is supported throughout the specification, but specifically the reasoning for choosing the most significant bit can be found on Col. 7, lines 37-47, of Patent No. 5,430,864 (parent application). The cited text points to the predictability of the behavior of the most significant bit (i.e. not be expected to change at the low levels of addition or subtraction to be expected in address manipulations on the stack pointer) as one incentive for utilizing the most significant bit as an indicator.

Notwithstanding the first sentence of the referenced paragraph, one skilled in the art would realize that the most significant bit is not the only bit with a predictable behavior characteristic since stack frames on a SPARC architecture, as on many other architectures, must be "aligned on four byte boundaries" (Col. 8, lines 40-44). One skilled in the art would further realize that aligning the stack frames on four byte boundaries means that the least significant bit also does not change in value as the stack frames are allocated and deallocated. Therefore, a person skilled in the art would realize that this allows the least significant bit to be used to indicate whether a set of data values are of a first word size or a second word size. One skilled in the art would know that this has the further advantage that a reference made incorrectly (*e.g.*, assuming a 64-bit stack pointer when it is really a 32-bit stack pointer, or vice versa) will result in an alignment exception, allowing incorrectly executing programs to be caught and terminated.

Therefore, one skilled in the art would be able to make and use the inventions of Claims 17-38 based on the specification without any undue experimentation. Accordingly, Applicants submit that the specification is enabling for context switching by testing a least significant bit of a stack pointer register.

C. Group II: Rejection of Claims 33-38 as Being Incomplete for Omitting Essential Steps

The Examiner argues that Claims 33-38 are incomplete for omitting an essential step. Specifically, the Examiner is of the opinion that "setting a width indication bit" is essential because otherwise there is no indication of a second word size in the first stack save area of the memory. Applicants disagree with the Examiner's conclusion.

Although the specification does discuss setting an indication in the first stack save area, the specification clearly notes that this is in accordance with an embodiment in which the indication is set in the most significant bit (Col. 7, lines 14-47). As stated in the Declaration and Power of Attorney for Reissue Patent Application submitted with the present reissue application, one skilled in the art would realize that the required steps necessary to perform the inventions of Claims 17-38 are reduced when the least significant bit is utilized rather than the most significant bit. Specifically, when the least significant bit of the stack pointer register is used to indicate the second word size, there is no need to transfer a least significant portion of a stack pointer value from the stack pointer register to the first stack save area in memory; or to set a width indication bit in the first stack save area in memory; or to transfer the stack pointer value from the stack pointer register to the second stack save area; or to know that the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

In light of the foregoing, Applicants submit that "setting a width indication bit" is not a necessary step when the least significant bit is used as in Claims 17-38. As such, the fact that the "setting a width indication bit" step is absent from Claims 33-38 should not preclude patentability.

IX. CONCLUSION AND RELIEF

Based on the foregoing, Applicants request that the Board overturn the rejection of all pending claims and hold that all of the claims of the present application are allowable.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN

Dated: 2/23/01

Raul D Mart

Raul D. Martinez
Reg. No. 46,904

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025
(310) 207-3800

CERTIFICATE OF MAILING:

I hereby certify that this correspondence is being deposited as First Class Mail with the United States Postal Service in an envelope addressed to: Assistant Commissioner for Patents, Washington, D.C. 20231 on February 23, 2001.

Laura Harmon

Laura Harmon Date 2/23/01

X. APPENDIX

The claims involved in this Appeal are as follows:

1. A method for context switching a processor that executes procedures having differing word sizes, comprising the steps of:

testing a most significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor each have a first word size or a second word size wherein the first word size is less than the second word size;

transferring the data values from a least significant portion of each register to a first stack save area in memory and transferring a least significant portion of a stack pointer value from the stack pointer register to the first stack save area in memory if the most significant bit of the stack pointer register indicates the first word size;

setting a width indication bit in the first stack save area in memory, and transferring the data values from the registers to a second stack save area in memory and transferring the stack pointer value from the stack pointer register to the second stack save area if the most significant bit of the stack pointer register indicates the second word size such that the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

2. The method of claim 1, further comprising the steps of:

testing the width indication bit in the first stack save area in memory;

reading the data values and the least significant portion of the stack pointer value from the first stack save area, and storing the data values into the least significant portions of the registers and storing the least significant portion of the stack pointer value into the stack pointer register, and clearing the most significant bit of the stack pointer register to indicate that the data values for the procedure have the first word size if the width indication bit in the first stack save area in memory does not indicate that the data values for the procedure have the second word size;

reading the data values and the stack pointer value from the second stack save area, and storing the data values into the registers and storing the stack pointer value in the stack pointer register if the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

3. The method of claim 2, wherein the width indication bit in the first stack save area comprises a least significant bit in a location of the first stack save area allocated to the stack pointer value for the procedure.

4. The method of claim 2, wherein the first stack save area is specified by the stack pointer value in the stack pointer register.

5. The method of claim 4, wherein the second stack save area is specified by the stack pointer value in the stack pointer register plus an offset value that corresponds to an area in memory required for the first stack save area.

6. The method of claim 5, wherein the first word size comprises 32 bits and the second word size comprises 64 bits.

7. The method of claim 6, wherein the registers in the processor and the stack pointer register in the processor comprise 16 registers each comprising 64 bits.

8. The method of claim 7, wherein the offset value and the area in memory for the first stack save area each comprise 16 multiplied by 4 bytes per register.

9. A processor that executes procedures having differing word sizes, comprising:

means for testing a most significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor each have a first word size or a second word size wherein the first word size is less than the second word size;

means for transferring the data values from a least significant portion of each register to a first stack save area in memory and transferring a least significant portion of a stack pointer value from the stack pointer register to the first stack save area in memory if the most significant bit of the stack pointer register indicates the first word size;

means for setting a width indication bit in the first stack save area in memory, and transferring the data values from the registers to a second stack save area in memory and transferring the stack pointer value from the stack pointer register to the second stack save area if the most significant bit of the stack pointer register indicates the second word size such that the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

10. The processor of claim 9, further comprising:

means for testing the width indication bit in the first stack save area in memory;

means for reading the data values and the least significant portion of the stack pointer value from the first stack save area, and storing the data values into the least significant portions of the registers and storing the least significant portion of the stack pointer value into the stack pointer register, and clearing the most significant bit of the stack pointer register to indicate that the data values for the procedure have the first word size if the width indication bit in the first stack save area in memory does not indicate that the data values for the procedure have the second word size;

means for reading the data values and the stack pointer value from the second stack save area, and storing the data values into the registers and storing the stack pointer value in the stack pointer register if the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

11. The processor of claim 10, wherein the width indication bit in the first stack save area comprises a least significant bit in a location of the first stack save area allocated to the stack pointer value for the procedure.

12. The processor of claim 10, wherein the first stack save area is specified by the stack pointer value in the stack pointer register.

13. The processor of claim 12, wherein the second stack save area is specified by the stack pointer value in the stack pointer register plus an offset value that corresponds to an area in memory required for the first stack save area.

14. The processor of claim 13, wherein the first word size comprises 32 bits and the second word size comprises 64 bits.

15. The processor of claim 14, wherein the registers in the processor and the stack pointer register in the processor comprise 16 registers each comprising 64 bits.

16. The processor of claim 15, wherein the offset value and the area in memory for the first stack save area each comprise 16 multiplied by 4 bytes per register.

17. A method for context switching a processor that executes procedures having differing word sizes, comprising the steps of:

testing a least significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor each have a first word size or a second word size wherein the first word size is less than the second word size;

transferring the data values from a least significant portion of each register to a first stack save area in memory and transferring a least significant portion of a stack pointer value from the stack pointer register to the first stack save area in memory if the least significant bit of the stack pointer register indicates the first word size;

setting a width indication bit in the first stack save area in memory, and transferring the data values from the registers to a second stack save area in memory and transferring the stack pointer value from the stack pointer register to the second stack save area if the least significant bit of the stack pointer register indicates the second word size such that the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

18. The method of claim 17, further comprising the steps of:

testing the width indication bit in the first stack save area in memory;

reading the data values and the least significant portion of the stack pointer value from the first stack save area, and storing the data values into the least significant portions of the registers and storing the least significant portion of the stack pointer value into the stack pointer register, and clearing the least significant bit of the stack pointer register to indicate that the data values for the procedure have the first word size if the width indication bit in the first stack save area in memory does not indicate that the data values for the procedure have the second word size;

reading the data values and the stack pointer value from the second stack save area, and storing the data values into the registers and storing the stack pointer value in the stack pointer register if the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

19. The method of claim 18, wherein the width indication bit in the first stack save area comprises a least significant bit in a location of the first stack save area allocated to the stack pointer value for the procedure.

20. The method of claim 18, wherein the first stack save area is specified by the stack pointer value in the stack pointer register.

21. The method of claim 20, wherein the second stack save area is specified by the stack pointer value in the stack pointer register plus an offset value that corresponds to an area in memory required for the first stack save area.

22. The method of claim 21, wherein the first word size comprises 32 bits and the second word size comprises 64 bits.

23. The method of claim 22, wherein the registers in the processor and the stack pointer register in the processor comprise 16 registers each comprising 64 bits.

24. The method of claim 23, wherein the offset value and the area in memory for the first stack save area each comprise 16 multiplied by 4 bytes per register.

25. A processor that executes procedures having differing word sizes, comprising:

means for testing a least significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor each have a first word size or a second word size wherein the first word size is less than the second word size;

means for transferring the data values from a least significant portion of each register to a first stack save area in memory and transferring a least significant portion of a stack pointer value from the stack pointer register to the first stack save area in memory if the least significant bit of the stack pointer register indicates the first word size;

means for setting a width indication bit in the first stack save area in memory, and transferring the data values from the registers to a second stack save area in memory and transferring the stack pointer value from the stack pointer register to the second stack save area if the least significant bit of the stack pointer register indicates the second word size such that the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

26. The processor of claim 25, further comprising:

means for testing the width indication bit in the first stack save area in memory;

means for reading the data values and the least significant portion of the stack pointer value from the first stack save area, and storing the data values into the least significant portions of the registers and storing the least significant portion of the stack pointer value into the stack pointer register, and clearing the least significant bit of the stack pointer register to indicate that the data values for the procedure have the first word size if the width indication bit in the first stack save area in memory does not indicate that the data values for the procedure have the second word size;

means for reading the data values and the stack pointer value from the second stack save area, and storing the data values into the registers and storing the stack pointer value in the stack pointer register if the width indication bit in the first stack save area in memory indicates that the data values for the procedure have the second word size.

27. The processor of claim 26, wherein the width indication bit in the first stack save area comprises a least significant bit in a location of the first stack save area allocated to the stack pointer value for the procedure.

28. The processor of claim 26, wherein the first stack save area is specified by the stack pointer value in the stack pointer register.

29. The processor of claim 28, wherein the second stack save area is specified by the stack pointer value in the stack pointer register plus an offset value that corresponds to an area in memory required for the first stack save area.

30. The processor of claim 29, wherein the first word size comprises 32 bits and the second word size comprises 64 bits.

31. The processor of claim 30, wherein the registers in the processor and the stack pointer register in the processor comprise 16 registers each comprising 64 bits.

32. The processor of claim 31, wherein the offset value and the area in memory for the first stack save area each comprise 16 multiplied by 4 bytes per register.

33. A method for context switching a processor that executes procedures having differing word sizes, comprising the steps of:

testing a least significant bit of a stack pointer register in the processor that indicates whether a set of data values for a procedure that are stored in a set of registers in the processor, each have a first word size or a second word size, wherein the first word size is less than the second word size;

transferring the data values from a least significant portion of each register to a first stack save area in memory if the least significant bit of the stack pointer register indicates the first word size;

transferring the data values from the registers to a second stack save area in memory if the least significant bit of the stack pointer register indicates the second word size.

34. The method of claim 33, wherein the first stack save area is specified by a stack pointer value in the stack pointer register.

35. The method of claim 34, wherein the second stack save area is specified by the stack pointer value in the stack pointer register plus an offset value that corresponds to an area in memory required for the first stack save area.

36. The method of claim 33, wherein the first word size comprises 32 bits and the second word size comprises 64 bits.

37. The method of claim 33, wherein the registers in the processor and the stack pointer register in the processor comprise 16 registers each comprising 64 bits.

38. The method of claim 35, wherein the offset value and the area in memory for the first stack save area each comprise 16 multiplied by 4 bytes per register.